- What were the decisions taken in the modelling?

  During the modeling process of the Amazon database, we had to make important decisions regarding the nodes and their relationships. This involved considering both our initial concept for the database and the need for a viable model that would allow for efficient querying later on. All of the relationships within the database are one-way, with customers placing orders that include a specified quantity. The quantity indicates the number of books within each order, which are further categorized by language, genre, and character. Additionally, each book is authored by one or more writers and published by one or more publishers. This forms the foundation of our overall database model.

- Why were these decisions taken?

  We have designed this model to be easily understandable for anyone. It expresses the logic behind all operations in the bookshop in a clear and logical manner, and is simple and easy to use when updating or querying.

- What were the consequences of these decisions?

  Because of the way we designed the model it was relatively easy to create the queries needed to modify and query the data in neo4j. This was also translated when we had to write the queries and mutations for GraphQL.

- What were the difficult and easy parts of the exercise?

  Easy:

    - Translating the Schema from neo4j to GraphQL
    - Creating the queries

  Difficult:

    - Setting up neo4j and Apollo GraphQL

- How does that compare to the other exercises?

  Compared to the second assignment this assignment was comparatively much easier to do and understand.

- What are the advantages and disadvantages of graph databases compared to the other database types?

  Advantages:

    - The structures are agile and flexible.

- The representation of relationships between entities is explicit.
- Queries output real-time results. The speed depends on the number of relationships.

Disadvantages:

- There is no standardized query language. The language depends on the platform used.
- Graphs are inappropriate for transactional-based systems.
- The user-base is small, making it hard to find support when running into a problem.